

② 12

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号

特開2004-29992

(P2004-29992A)

(43) 公開日 平成16年1月29日(2004.1.29)

(51) Int. Cl. ⁷	F I	テーマコード (参考)
G05B 23/02	G05B 23/02 302Y	5B042
G05B 19/048	G06F 1/04 302Z	5B045
G06F 1/04	G06F 9/46 330C	5B098
G06F 9/46	G06F 9/46 340B	5H220
G06F 11/30	G06F 11/30 320D	5H223
審査請求 未請求 請求項の数 14 O L (全 12 頁) 最終頁に続く		

(21) 出願番号	特願2002-182424 (P2002-182424)	(71) 出願人	000004260
(22) 出願日	平成14年6月24日 (2002.6.24)		株式会社デンソー
		(74) 代理人	100068755
			弁理士 恩田 博宣
		(74) 代理人	100105957
			弁理士 恩田 誠
		(72) 発明者	宮地 和孝
			愛知県刈谷市昭和町1丁目1番地 株式会
			社デンソー内
		(72) 発明者	小林 稔昌
			愛知県刈谷市昭和町1丁目1番地 株式会
			社デンソー内
		Fターム(参考)	5B042 JJ13 JJ15 JJ19 JJ21 JJ26
			JJ29
			最終頁に続く

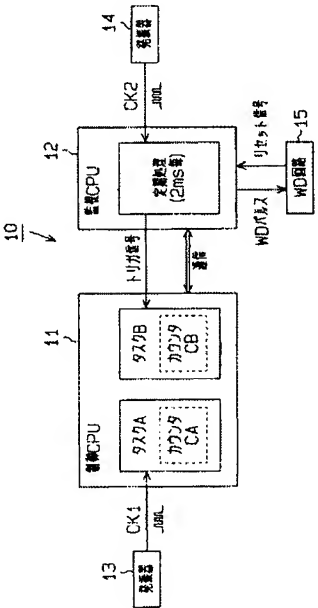
(54) 【発明の名称】 電子制御装置

(57) 【要約】

【課題】 クロック異常や処理抜け異常を適正に検出し、ひいてはCPUの信頼性の確保を図ること。

【解決手段】 エンジンECU10は、制御CPU11と監視CPU12とを備える。監視CPU12は自身に入力されるクロックを基に一定周期のトリガ信号を生成し制御CPU11に送信する。また、制御CPU11は、自身に入力されるクロックを基に一定周期でタスクAを起床し、カウンタCAをカウントすると共に、監視CPU12より入力されるトリガ信号を基にタスクBを起床し、カウンタCBをカウントする。そして、これらカウンタCA及びCBの差分を、前後する2つのタイミングで比較し、その比較結果からクロック異常を検出する。

【選択図】 図1



【特許請求の範囲】**【請求項1】**

第1CPUと第2CPUとを備え、これら各CPUに個別のクロック信号が入力されると共に該クロック信号を基に各CPUで定期的な演算処理が実施され、前記第2CPUは自身に入力されるクロック信号を基に一定周期の時間情報を前記第1CPUに通知する電子制御装置であって、

前記第1CPUは、自身に入力されるクロック信号を基に一定周期で主カウンタをカウントすると共に、前記第2CPUより入力される一定周期の時間情報を基に副カウンタをカウントし、これら主副両カウンタの差分を、前後する2つのタイミングで比較し、その比較結果から前記クロック信号の異常を検出することを特徴とする電子制御装置。

【請求項2】

前記第1CPUは、一定周期でタスクAを起床し、当該タスクAにて主カウンタをカウントすると共に、前記第2CPUからの時間情報をトリガとしてタスクBを起床し、当該タスクBにて副カウンタをカウントする請求項1記載の電子制御装置。

【請求項3】

前記第2CPUは、クロック信号に基づく定期処理にて一定周期でトリガ信号を出力し、前記第1CPUは、該トリガ信号の入力の都度、副カウンタをカウントする請求項1又は2記載の電子制御装置。

【請求項4】

前記第2CPUは、クロック信号に基づく定期の通信処理にて一定周期でデータ送信を行い、前記第1CPUは、第2CPUからのデータ受信の都度、副カウンタをカウントする請求項1又は2記載の電子制御装置。

【請求項5】

前記第2CPUより所定周期で反転するウォッチドッグパルスを入力し、そのウォッチドッグパルスが所定時間以上反転しないと第2CPUに対してリセット信号を出力する監視回路を更に備え、前記第1CPUは、前記ウォッチドッグパルスを第2CPUより入力し、その入力の都度、副カウンタをカウントする請求項1又は2記載の電子制御装置。

【請求項6】

前記第1CPUは、前後2つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が毎回同一である場合に、クロック異常である旨判定する請求項1乃至5の何れかに記載の電子制御装置。

【請求項7】

前記第1CPUは、前後2つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が一方のカウンタのカウント分相当である場合に、何れかのカウンタが停止している旨判定する請求項1乃至6の何れかに記載の電子制御装置。

【請求項8】

主副両カウンタの周期が同一である請求項1乃至7の何れかに記載の電子制御装置。

【請求項9】

前記第1CPUは、前記第2CPUより通知される時間情報をトリガとして所定のタスクを起床し、当該タスク内において規定の処理単位で処理終了の履歴を残し、次のタスク起床時に前記履歴から処理抜け異常を検出する請求項1乃至8の何れかに記載の電子制御装置。

【請求項10】

請求項9記載の電子制御装置において、前記第1CPUは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残す電子制御装置。

【請求項11】

請求項9又は10記載の電子制御装置において、前記第1CPUは、前記第2CPUからの時間情報にて起床される所定のタスクの最後に、前記副カウンタをカウントする電子制御装置。

【請求項12】

第1CPUと第2CPUとを備え、これら各CPUに個別のクロック信号が入力されると共に該クロック信号を基に各CPUで定期的な演算処理が実施され、前記第2CPUは自身に入力されるクロック信号を基に一定周期の時間情報を前記第1CPUに通知する電子制御装置であって、

前記第1CPUは、前記第2CPUより通知される時間情報をトリガとして所定のタスクを起床し、当該タスク内において規定の処理単位で処理終了の履歴を残し、次のタスク起床時に前記履歴から処理抜け異常を検出することを特徴とする電子制御装置。

【請求項13】

請求項12記載の電子制御装置において、前記第1CPUは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残す電子制御装置。

【請求項14】

前記第1CPUは電子スロットル制御を実施し、前記第2CPUは電子スロットル制御に関するフェイルセーフ処理を実施する車両用電子制御装置として適用され、前記第1CPUは、クロック異常又は処理抜け異常である旨を検出した時にそれを第2CPUに通知し、第2CPUは、第1CPUからの異常情報に基づきフェイルセーフ処理を実施する請求項1乃至13の何れかに記載の電子制御装置。

【発明の詳細な説明】**【0001】****【発明の属する技術分野】**

本発明は、2つ以上のCPUを有する電子制御装置にかかり、特に各CPUに供給されるクロック信号の異常や処理抜け異常を好適に検出するための電子制御装置に関するものである。

【0002】**【従来の技術】**

車両制御等を実施する電子制御装置として、複数のCPUを備え、各CPU間で相互に異常を監視する機能を持つものがある。その一つとして、所定周期で反転するウォッチドッグパルス(WDパルス)をモニタしてCPUの動作状態を監視する方法がある。これは、一方のCPUが、他方のCPUよりWDパルスを入力し、該WDパルスが停止すると当該他方のCPUが異常である(暴走している)と判定するものである。

【0003】**【発明が解決しようとする課題】**

この種の電子制御装置に関わる異常として、発振器等から各CPUに供給されるクロック信号の異常がある。この場合、発振器の異常等によりクロック信号の周波数が大小変化すると、そのクロック異常の状態によってはその異常が検出できないことが考えられる。

【0004】

例えば、クロック信号の周波数が正常時よりも大きくなる場合、それに起因してCPU内の時間演算の精度が低下する。そのため、定期的の実施されるべき各種の演算処理が所定の周期で実施されなくなるという不都合が生じる。かかる場合、上記の如くWDパルスによりCPUの異常検出を実施しても、WDパルスは定期的に行われているため、クロック異常を判定することができない。

【0005】

また、優先度の異なる複数のタスクを起床するCPUでは、CPUの処理負荷が高まると低優先度のタスクで処理抜けが発生する場合が考えられる。この場合、通常WDパルスの発行は高優先度の処理であるため、低優先度処理が抜けたときでも正常にWDパルスが発行される。そのため、処理抜けが発生してもWDパルスには影響が出ず、処理抜けを検出することができない。最悪の場合、そのまま処理を継続する可能性が考えられる。

【0006】

本発明は上記問題に着目してなされたものであって、その目的とするところは、クロック

異常や処理抜け異常を適正に検出し、ひいてはCPUの信頼性の確保を図ることができる電子制御装置を提供することである。

【0007】

【課題を解決するための手段】

請求項1に記載の発明では、第1CPUと第2CPUとを備え、これら各CPUに個別のクロック信号が入力されると共に該クロック信号を基に各CPUで定期的な演算処理が実施される。前記第2CPUは自身に入力されるクロック信号を基に一定周期の時間情報を前記第1CPUに通知する。また特に、前記第1CPUは、自身に入力されるクロック信号を基に一定周期で主カウンタをカウントすると共に、前記第2CPUより入力される一定周期の時間情報を基に副カウンタをカウントする。そして、これら主副両カウンタの差分を、前後する2つのタイミングで比較し、その比較結果から前記クロック信号の異常を検出する。

【0008】

仮に第2CPU側のクロック信号が異常となり、その周波数が正常時よりも小さくなる場合、第2CPUから第1CPUへ通知される時間情報の時間間隔が広がり、結果として副カウンタのカウント動作が遅くなる。この場合、主副両カウンタの差分が正常時とは相違し、更に前後する差分を比較することで、クロック信号が異常である旨判定できる。つまり、クロック信号の周期が変化するような異常を検出することができる。その結果、クロック異常を適正に検出し、ひいてはCPUの信頼性の確保を図ることができるようになる。

【0009】

請求項2に記載の発明では、前記第1CPUは、一定周期でタスクAを起床し、当該タスクAにて主カウンタをカウントすると共に、前記第2CPUからの時間情報をトリガとしてタスクBを起床し、当該タスクBにて副カウンタをカウントする。この場合、タスクA及びタスクBが適宜起床されることで主副両カウンタのカウント動作を行わせることができる。

【0010】

上記の如く第1CPUは、第2CPUから時間情報の通知を受けて副カウンタを動作させるが、かかる場合において、時間情報の通知を以下の形態で実現すると良い。

・請求項3に記載の発明では、前記第2CPUは、クロック信号に基づく定期処理にて一定周期でトリガ信号を出力し、前記第1CPUは、該トリガ信号の入力の都度、副カウンタをカウントする。

・請求項4に記載の発明では、前記第2CPUは、クロック信号に基づく定期の通信処理にて一定周期でデータ送信を行い、前記第1CPUは、第2CPUからのデータ受信の都度、副カウンタをカウントする。

・請求項5に記載の発明では、前記第2CPUより所定周期で反転するウォッチドッグパルスを入力し、そのウォッチドッグパルスが所定時間以上反転しないと第2CPUに対してリセット信号を出力する監視回路を更に備え、前記第1CPUは、前記ウォッチドッグパルスを第2CPUより入力し、その入力の都度、副カウンタをカウントする。

【0011】

上記何れの場合にも、第1CPUにおいて一定周期で副カウンタがカウントされ、その副カウンタと主カウンタとの差分によりクロック異常が適正に検出できる。

【0012】

請求項6に記載の発明では、前記第1CPUは、前後2つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が毎回同一である場合に、クロック異常である旨判定する。これにより、クロック異常が精度良く検出できる。

【0013】

請求項7に記載の発明では、前記第1CPUは、前後2つのタイミングの主副両カウンタの差分を比較して差分変化量を求め、その差分変化量が一方のカウンタのカウント分相当である場合に、何れかのカウンタが停止している旨判定する。これにより、カウンタ停止

の異常が精度良く検出できる。請求項2のように、タスクAで主カウンタがカウントされ、タスクBで副カウンタがカウントされる場合、何れかのタスクが停止したことが精度良く検出できる。

【0014】

請求項8に記載したように、主副両カウンタの周期が同一である場合、それら両カウンタが共に正常であれば、差分が常に一致する。それ故、主副両カウンタの差分の比較が容易に実施できる。

【0015】

請求項9に記載の発明では、前記第1CPUは、前記第2CPUより通知される時間情報をトリガとして所定のタスクを起床し、当該タスク内において規定の処理単位で処理終了の履歴を残す。そして、次のタスク起床時に前記履歴から処理抜け異常を検出する。この場合、前記タスクが最後まで実施されれば、全処理について処理終了の履歴が残るが、途中までしか実施されなければ未実施分については処理終了の履歴が残らない。従って、当該タスクについて処理抜けを検出することができる。また、履歴を確認することにより、処理抜けが発生した位置を特定することができる。

【0016】

上記請求項9の発明では請求項10に記載したように、前記第1CPUは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残すと良い。優先度の異なる複数のタスクが存在する場合、優先度の低いタスクは、その実施途中で高優先度のタスクが割り込むことで処理抜け（処理の中断）が生じるおそれがあるが、かかる場合に処理抜けの発生やその発生場所の特定が可能となる。

【0017】

また、請求項11に記載したように、前記第1CPUは、前記第2CPUからの時間情報にて起床される所定のタスクの最後に、前記副カウンタをカウントすると良い。これにより、処理抜けが発生した場合に副カウンタがカウントされなくなり、主副両カウンタの差分から異常発生の判定が可能となる。

【0018】

一方、請求項12に記載の発明でも、上記請求項9と同様に、前記タスクが最後まで実施されれば、全処理について処理終了の履歴が残るが、途中までしか実施されなければ未実施分については処理終了の履歴が残らない。従って、当該タスクについて処理抜けを検出することができる。また、履歴を確認することにより、処理抜けが発生した位置を特定することができる。

【0019】

更に、請求項13に記載したように、前記第1CPUは、優先度の異なる複数のタスクを起床し、そのうち優先度の低いタスクにて規定の処理単位で処理終了の履歴を残すことで、高優先度のタスクが割り込んで処理抜け（処理の中断）が発生しても、当該処理抜けの発生やその発生場所の特定が可能となる。

【0020】

また、請求項14に記載の発明は、前記第1CPUは電子スロットル制御を実施し、前記第2CPUは電子スロットル制御に関するフェイルセーフ処理を実施する車両用電子制御装置として適用される。そして、前記第1CPUは、クロック異常又は処理抜け異常である旨を検出した時にそれを第2CPUに通知し、第2CPUは、第1CPUからの異常情報に基づきフェイルセーフ処理を実施する。この場合、クロック異常や処理抜け異常が発生しても、その適正なフェイルセーフ処理が実施できる。

【0021】

【発明の実施の形態】

以下、本発明を具体化した一実施の形態を図面に従って説明する。本実施の形態では、車両に搭載されるエンジンECUとして本発明の電子制御装置を具体化している。図1は、本実施の形態におけるエンジンECUの構成を示すブロック図である。

【0022】

図1において、エンジンECU10は、エンジンの噴射制御、点火制御及び電子スロットル制御を実施するための制御CPU11と、電子スロットル制御に関するフェイルセーフ処理や制御CPU11の監視を実施するための監視CPU12とを備える。これらの各CPU11、12は相互に通信可能に接続されている。制御CPU11には発振器13よりクロックCK1（クロック信号）が入力され、監視CPU12には発振器14よりクロックCK2（クロック信号）が入力される。なお本実施の形態では、制御CPU11が「第1CPU」に相当し、監視CPU12が「第2CPU」に相当する。

【0023】

制御CPU11は、発振器13からのクロックCK1に基づき一定周期でタスクAを起床し、そのタスクA内にて毎回カウンタCAをカウントアップする。タスクAは高優先度のタスクであり、本実施の形態では2ms毎に起床されるようになっている。また、制御CPU11には監視CPU12からトリガ信号が取り込まれる。制御CPU11はこのトリガ信号に基づきタスクBを起床し、そのタスクB内にて毎回カウンタCBをカウントアップする。

【0024】

一方、監視CPU12は、発振器14からのクロックCK2に基づき2ms毎に定期処理を起床し、その定期処理にて一定周期のトリガ信号を生成し出力する。このトリガ信号が制御CPU11に取り込まれる。本実施の形態では、カウンタCAが「主カウンタ」に、カウンタCBが「副カウンタ」に相当し、トリガ信号が「一定周期の時間情報」に相当する。

【0025】

また、監視CPU12は「監視回路」としてのWD回路15に対してWDパルスを出力し、WD回路15は監視CPU12からのWDパルスが所定時間以上反転しなかった場合に監視CPU12に対してリセット信号を出力する。

【0026】

ここで、カウンタCA及びCBの動作を図2にて説明する。カウンタCA及びCBが共に正常動作する場合、図2の(a)に示すように、各カウンタが何れも2ms毎にカウントアップされる。この場合、カウンタCA及びCBの差分を ΔC_n ($=CA - CB$)とすると、タイミング t_1 ではその差分が ΔC_1 、タイミング t_2 ではその差分が ΔC_2 となる。図2の(a)は正常動作を表すため $\Delta C_1 = \Delta C_2$ となる。

【0027】

これに対し、図2の(b)～(d)はカウンタCBが異常動作となる場合を示す。(b)～(d)を詳しく説明する。(b)は、カウンタCBの周期が正常時よりも大きくなっている。これは監視CPU12側のクロックCK2が異常（クロック異常）となり、それが原因で定期処理の起床周期（トリガ信号の周期）が変わり、更にはカウンタCBの周期が変化したと考えられる。この場合、タイミング t_1 、 t_2 での差分 ΔC_1 、 ΔC_2 は $\Delta C_1 < \Delta C_2$ となる。

【0028】

また、(c)は、カウンタCBが停止する異常を示し、(d)は、監視CPU12での定期処理が抜けてしまう異常（処理抜け異常）を示す。なお、処理抜けは不定期に発生し、この処理抜けによりカウンタCBが一定周期でカウントアップされない事態が生じる。これら(c)、(d)の場合にもやはり、タイミング t_1 、 t_2 での差分 ΔC_1 、 ΔC_2 が $\Delta C_1 < \Delta C_2$ となる。

【0029】

本実施の形態では、上記図2の(b)～(d)の各異常について何れも検出可能であって、更に各異常の形態を特定できる異常検出手法を提案する。上記図2の(b)の場合、カウンタCA及びCBのカウントアップの周期が相違するため各カウンタ値の差分 ΔC_n は毎回相違するが、同 ΔC_n の変化は一律であり、一定の時間間隔で前後する2つの ΔC_n を比較すると、その変化量は一定である。つまり、前後する差分の変化量を DC_n （以下、差分変化量 DC_n ）とすると、その差分変化量 DC_n は毎回同じ値となる

($DC_n = DC_{n-1}$ となる)。それ故、その差分変化量 DC_n によりクロック異常が特定できる。勿論、カウンタCBの周期が小さくなる場合や、カウンタCAの周期が大きくなる(又は小さくなる)場合にも同様にクロック異常が特定できる。

【0030】

また、上記図2の(c)の場合、一方のカウンタCAは正常にカウントアップし、他方のカウンタCBは停止しているため、各カウンタCA及びCBの差分変化量 DC_n はカウンタCAのカウントアップ分に相当する。この場合、差分変化量 DC_n は、 $t_1 - t_2$ 間の時間間隔 DT_n をカウンタ1周期分の時間 T (2ms) で除算した値 (DT_n / T) に一致する ($DC_n = DT_n / T$ となる)。これにより、タスクB停止の異常が特定できる。

【0031】

また、上記図2の(d)の場合、上記(b)、(c)と同様に $\Delta C_1 \neq \Delta C_2$ (すなわち、 $DC_n \neq 0$) であるが、差分 ΔC_n の変化は不定期なものとなる。よって、 $\Delta C_1 \neq \Delta C_2$ であり、且つ上記図2の(b)、(c)に該当しなければ、処理抜け異常であると特定できる。

【0032】

次に、各CPU11, 12で起床される演算処理について図3～図6のフローチャートを参照しながら詳しく説明する。まずはじめに、図3はタスクAの手順を示すフローチャートであり、このタスクAは制御CPU11により2ms毎に起床される。

【0033】

図3において、先ずステップ101では、カウンタCA及びCBに基づいて異常検出処理を実施する。但しその詳細(図4の処理)は後述する。また、ステップ102では、タスクAに関して通常処理を実施し、続くステップ103では、カウンタCAを1インクリメントする。このタスクAにより、2ms毎にカウンタCAがカウントアップされるようになる。

【0034】

図4の異常検出処理では、8ms毎に一連の異常検出を実施することとしており、ステップ201がYESであることを条件に後続のステップ202に進む(つまり、タスクA起床の4回に1回の割合で異常検出が実施される)。

【0035】

ステップ202に進むと、カウンタCA及びCBの差分 ΔC_n がどれだけ変化したかを示す差分変化量 DC_n を算出する。具体的には、カウンタCA及びCBの差分の今回値 ΔC_n と前回値 ΔC_{n-1} との差を取り、差分変化量 DC_n を算出する ($DC_n = \Delta C_n - \Delta C_{n-1}$)。

【0036】

その後、ステップ203では、差分変化量 DC_n が0であるか否かを判別する。前記図2の(a)で説明した通りカウンタCA及びCBが何れも正常の場合、 $DC_n = 0$ ($\Delta C_n = \Delta C_{n-1}$) となる。 $DC_n = 0$ の場合、ステップ204に進み、各種の異常カウンタErrCk, ErrF, ErrStを何れも0にクリアする。そしてその後、本処理を一旦終了する。

【0037】

また、 $DC_n \neq 0$ の場合、ステップ205に進み、差分変化量の今回値 DC_n と前回値 DC_{n-1} とを比較する。そして、 $DC_n = DC_{n-1}$ であれば、クロック異常であると推定し、ステップ206でクロック異常カウンタErrCkを1インクリメントする。これは、前記図2の(b)の場合に該当する。

【0038】

また、 $DC_n \neq DC_{n-1}$ の場合、ステップ207で時間間隔 DT_n (前回の異常検出から今回の異常検出までの時間幅) を算出し、続くステップ208で $DC_n = DT_n / T$ であるか否かを判別する。そして、YESの場合、タスクBの停止異常であると推定し、ステップ209でタスクB停止カウンタErrStを1インクリメントする。ま

た、NOの場合、処理抜け異常であると推定し、ステップ210で処理抜け異常カウンタErrFを1インクリメントする。ステップ208がYESであることは前記図2の(c)の場合に該当し、同ステップ208がNOであることは前記図2の(d)の場合に該当する。

【0039】

その後、ステップ211～213では、各異常カウンタErrCk, ErrSt, ErrFが所定の判定値Lmtよりも小さいか否かを判別する。そして、何れも判定値Lmt未満であれば、そのまま本処理を終了する。また、判定値Lmt以上となる異常については、それに対応する異常フラグ(XErrCk, XErrSt, XErrFの何れか)をONする(ステップ214～216)。なお、判定値Lmtは、何れも同一にする必要はなく、各異常カウンタErrCk, ErrSt, ErrF毎に個別に設定することも可能である。また、同一の異常が所定回連続して検出された場合にのみ、異常フラグをONする構成としても良い。

【0040】

また、図5はタスクBの手順を示すフローチャートであり、このタスクBは、監視CPU12からトリガ信号を入力する都度制御CPU11により起床される。実際には、トリガ信号は2ms周期のパルス信号であり、タスクAと同じ周期でこのタスクBが実施されるようになっている。因みに、トリガ信号は、図6に示す定期処理にて生成される。つまり、監視CPU12は、定期処理内の通常処理(ステップ401, 403)の途中でトリガ信号を生成し出力する(ステップ402)。

【0041】

タスクBでは、n個の処理(処理1～処理n)が順次実施され、各処理が実施される都度、処理完了を表す処理完了フラグXfinがONされる。つまりこのとき、処理1～処理nが規定の処理単位であり、その処理終了の履歴が処理完了フラグXfinとして残される。そして、次のタスクB起床時においてこの処理完了フラグXfinに基づき、処理抜け場所が特定されるようになっている。

【0042】

図5において、先ずステップ301では、処理番号を示すiを0にクリアし、続くステップ302では、処理iの処理完了フラグxfi(i)がONしているか否かを判別する。フラグONであれば、前回のタスクBで当該処理iが処理完了したと判断される。この場合、ステップ303でi=nであるか否かを判別する。そして、i=nが成立するまで、ステップ304でiを1ずつ加算しつつ、フラグxfi(i)がONしているか否かを繰り返し判別する。

【0043】

フラグxfi(i)=OFFの場合には、その該当する処理iで処理抜けが発生したと判断できる。そのため、ステップ305に進み、その時の処理iを処理抜け発生位置としてメモリに記憶する。

【0044】

その後、ステップ306では、処理完了フラグxfiを全てクリアする。ステップ307～312では、処理1～処理nを順次実施すると共に、各処理の実施直後に、各々に対応する処理完了フラグXfinをONしていく。最後に、ステップ313では、カウンタCBを1インクリメントする。

【0045】

上記図4の異常検出処理にて異常である旨検出されると(前記異常フラグがONされると)、それが制御CPU11から監視CPU12に通知され、監視CPU12では、電子スロットル制御の通電カットなど、所定のフェイルセーフ処理が実施される。この場合、クロック異常や処理抜け異常が発生しても、その適正なフェイルセーフ処理が実施でき、エンジンECU10としての信頼性が確保される。

【0046】

以上詳述した本実施の形態によれば、以下に示す効果が得られる。

クロック異常が発生すると、カウンタC A及びC Bの差分 ΔC_n が正常時とは相違するため、前後する2つの差分 ΔC_n を比較することでクロック異常である旨判定できる。つまり、クロックの周期が変化するような異常を検出することができる。その結果、クロック異常を適正に検出し、ひいてはC P Uの信頼性の確保を図ることができるようになる。

【0047】

カウンタC A及びC Bの差分変化量 ΔC_n が毎回同一である場合にクロック異常である旨判定するため、クロック異常が精度良く検出できる。

また、カウンタC A及びC Bの差分変化量 ΔC_n が一方のカウンタのカウント分相当である場合に何れかのカウンタが停止している旨判定するため、カウンタ停止の異常が精度良く検出できる。またこれは、何れかのタスクが停止したことが精度良く検出できることと同意である。

【0048】

タスクB内において規定の処理単位で処理終了の履歴（処理完了フラグx f i n）を残し、次のタスク起床時に前記履歴から処理抜け異常を検出するため、処理抜けが発生した位置を確実に特定することができる。

【0049】

なお本発明は、上記以外に次の形態にて具体化できる。

上記実施の形態では、制御C P U 1 1は、監視C P U 1 2からトリガ信号を入力し、そのトリガ信号にてタスクBを起床したが、この構成を変更しても良い。例えば、

（1）制御C P U 1 1は、監視C P U 1 2から定期的にデータを受信する都度、タスクBを起床する。但しこの場合、上記2つのC P U間で定期的な通信が行われることが前提条件となる。

（2）制御C P U 1 1は、監視C P U 1 2からWDパルスを入力し、その入力の都度、タスクBを起床する。この場合、前記図1の構成において、監視C P U 1 2からWD回路1 5へのWDパルスを分岐させ、当該WDパルスを制御C P U 1 1にも入力する構成とすれば良い。

【0050】

これら何れの場合にも、制御C P U 1 1において一定周期でカウンタC Bがカウントされ、そのカウンタC A及びC Bの差分によりクロック異常が適正に検出できる。

【0051】

上記実施の形態では、カウンタC A及びC Bの周期（すなわち、タスクA及びタスクBの起床周期）が同一である場合を例示したが、その周期が相違するものであっても良い。カウンタC A及びC Bの周期が相違しても、各カウンタが何れも正常であればその差分 ΔC_n はある規則性を保つ。そのため、やはり前後する2つのタイミングで差分 ΔC_n を比較し、その比較結果からクロック異常が検出できる。

【0052】

本発明は、エンジンE C U以外にも他のE C Uに適用でき、更に車両用電子制御装置以外の用途に具体化することも可能である。要は、2つ以上のC P Uを備え、各C P Uに個別のクロック信号が入力されると共に該クロック信号を基に各C P Uで定期的な演算処理が実施される構成の電子制御装置であれば本発明が適用できる。

【図面の簡単な説明】

【図1】 発明の実施の形態におけるエンジンE C Uの構成を示すブロック図。

【図2】 （a）～（d）はカウンタ動作を示すタイムチャート。

【図3】 制御C P UによるタスクAの手順を示すフローチャート。

【図4】 制御C P Uによる異常検出処理を示すフローチャート。

【図5】 制御C P UによるタスクBの手順を示すフローチャート。

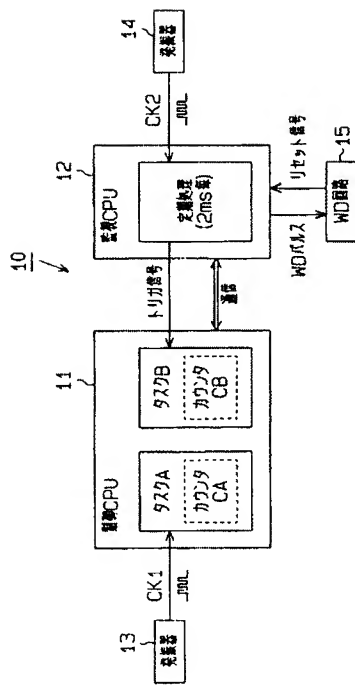
【図6】 監視C P Uによる定期処理を示すフローチャート。

【符号の説明】

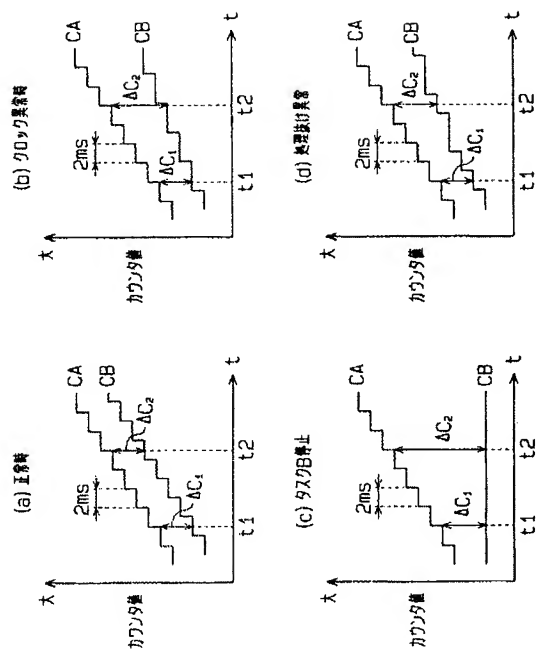
1 0…E C U、1 1…制御C P U、1 2…監視C P U、1 3、1 4…発振器、1 5…監視

回路。

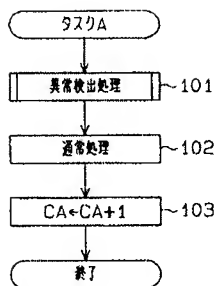
【図1】



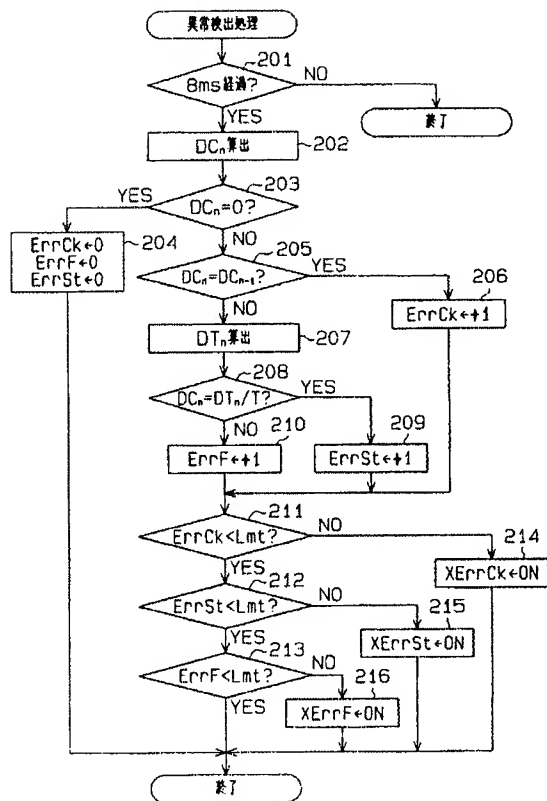
【図2】



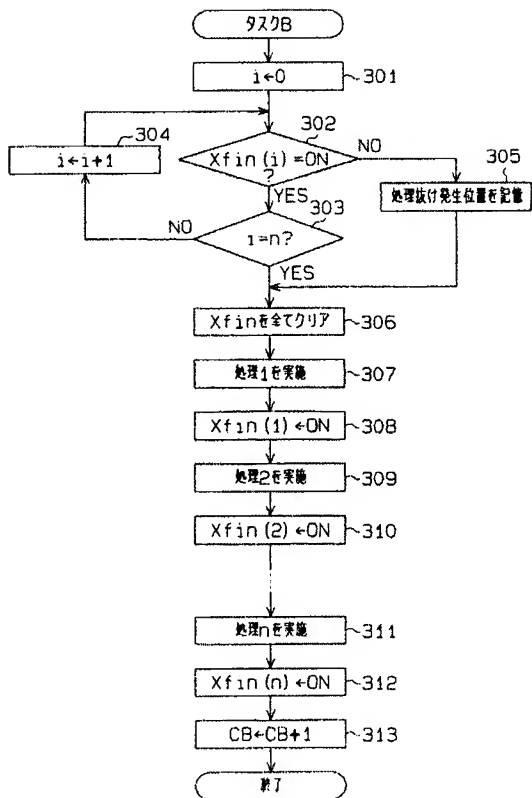
【図3】



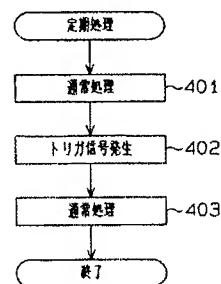
【図4】



【図5】



【図6】



(51)Int.Cl.⁷

G 0 6 F 15/177

F I

G 0 6 F 15/177 6 7 8 A

G 0 5 B 19/05 D

テーマコード (参考)

Fターム(参考) 5B045 JJ02 JJ13

5B098 AA10 BA04 GA02 GA04 GC06

5H220 BB09 CC07 CC09 CX01 EE10 HH03 JJ12 JJ26 KK01 LL01

5H223 AA10 CC03 DD03 EE04